

# Node-balancing by edge-increments

Friedrich Eisenbrand<sup>1</sup> and Shay Moran<sup>2,3</sup> and Rom Pinchasi<sup>2</sup> and Martin Skutella<sup>3</sup>

<sup>1</sup> EPFL, 1015 Lausanne, Switzerland

<sup>2</sup> Technion, Israel Institute of Technology, Haifa 32000, Israel.

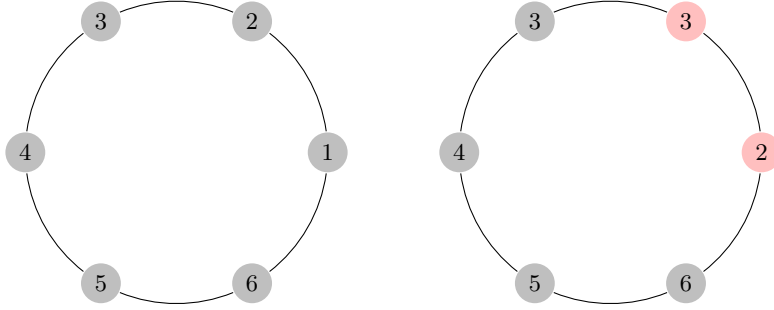
<sup>3</sup> Max Planck Institute for Informatics, Saarbrücken, Germany.

<sup>4</sup> Technische Universität Berlin, Straße des 17. Juni 136, 10623 Berlin, Germany

**Abstract.** Suppose you are given a graph  $G = (V, E)$  with a weight assignment  $w : V \rightarrow \mathbb{Z}$  and that your objective is to modify  $w$  using legal steps such that all vertices will have the same weight, where in each legal step you are allowed to choose an edge and increment the weights of its end points by 1.

In this paper we study several variants of this problem for graphs and hypergraphs. On the combinatorial side we show connections with fundamental results from matching theory such as Hall's Theorem and Tutte's Theorem. On the algorithmic side we study the computational complexity of associated decision problems.

Our main results are a characterization of the graphs for which any initial assignment can be balanced by edge-increments and a strongly polynomial-time algorithm that computes a balancing sequence of increments if one exists.



**Fig. 1.** The node-weights after one incrementing step.

## 1 Introduction

The following puzzle is often used as an introductory puzzle for the method of invariance and potential functions: Six boxes numbered 1 to 6 are arranged in

a cycle. For every  $1 \leq i \leq 6$ , we start with  $i$  oranges in box number  $i$ . At each step we are allowed to add one orange to each of two adjacent boxes. Prove that we will never be able to make all boxes contain the same number of oranges.

One of the simple solutions to this puzzle is to observe that the total number of oranges in boxes 1, 3, 5 is always smaller than the total number of oranges in boxes 2, 4, 6 and this never changes through each step of the game.

In this paper we consider the natural generalization of the puzzle above to arbitrary graphs. Let  $G = (V, E)$  be a finite graph, let  $w : V \rightarrow \mathbb{N}$  be a non-negative integer weight function on its vertices and let  $e = \{u, v\} \in E$ . A *positive step* on  $e$  modifies  $w$  by increasing the weights of  $u, v$  by 1 unit. We say that  $w$  is *equatable in  $G$*  if there exists a sequence of only positive steps,  $S = s_1, \dots, s_m$ , after which all vertices have the same weight. We also say that the sequence  $S$  *positively equates  $w$* .

Our *main results* are the following.

- i) We characterize those graphs  $G = (V, E)$  for which any initial assignment  $w : V \rightarrow \mathbb{N}_0$  is equatable. These are the connected graphs with an odd number of nodes for which  $G - U$  has less than  $|U|$  isolated vertices for any  $U \subset V$ . Here  $G - U$  is the subgraph of  $G$  that is induced by  $V \setminus U$ . (Theorem 1)
- ii) We show that the following problem can be solved in strongly polynomial time. Given a graph  $G = (V, E)$  and an initial assignment  $w : V \rightarrow \mathbb{N}_0$ , decide whether  $w$  is equatable and compute an equating multiset of edges. (Theorem 3)
- iii) An initial assignment  $w$  of the nodes of a bipartite graph  $G = (L + R, E)$  is not equatable if  $w(L) \neq w(R)$ , the difference  $w(L) - w(R)$  is invariant under edge-increments. However, each balanced assignment with  $w(L) = w(R)$  is equatable if and only if the strict Hall condition holds: For any nonempty set of vertices  $X$  that is properly contained in  $L$  or in  $R$ , one has  $|X| < |N(X)|$ . Here  $N(X)$  denotes the neighborhood of  $X$ . (Theorem 4)
- iv) Finally we show that the analog of the decision problem ii) is *NP*-hard for hypergraphs. (Theorem 5).

**Related work** The problem of equating the node-weights is closely related to *perfect  $b$ -matchings*, [15]. Let  $b \in \mathbb{N}_0^{|V|}$  be a vector of non-negative node-weights. A  $b$ -matching of a graph  $G = (V, E)$  is a vector  $x \in \mathbb{N}_0^{|E|}$  that satisfies

$$\sum_{e \in \delta(v)} x_e \leq b_v, \tag{1}$$

where  $\delta(v)$  denotes the set of edges of  $G$  that are incident to  $v$ . A  $b$ -matching is *perfect*, if the inequality in (1) can be replaced by equality. Thus  $b$ -matchings are a generalization of *matchings*, where  $b$  is the all ones vector.

What is the relationship between  $b$ -matchings and the process of equating positive weights in graphs by edge-increments? Suppose that the given initial

weight assignment  $w \in \mathbb{N}_0^{|V|}$  is equatable and that the resulting equated node-weight is  $\beta \in \mathbb{N}$ . Then, the edge-increments that lead to the balanced node-weight  $\beta$  are a  $b$ -matching  $x \in \mathbb{N}_0^{|E|}$  with  $b_v = \beta - w_v$  for each vertex  $v$ . By incrementing the node-weights of each edge  $e$  exactly  $x_e$  times, one arrives at a balanced assignment with weight  $\beta$  on all the nodes.

Maximum weight  $b$ -matchings can be computed in polynomial time [5,4,3]. The currently fastest algorithms for maximum weight matching are by Gabov [6], and Gabov and Tarjan [7]. The fastest algorithm for weighted  $b$ -matching is by Anstee [1]. Recent exciting progress for maximum cardinality matching has been given by Madry [13] improving upon the  $O(m\sqrt{n})$  running time of Hopcroft and Karp [10] and [12] in the sparse case.

A related notion to equatable graphs is the one of a *regularizable* graph. A graph is regularizable, if there exists a  $k$  and a perfect  $k$ -matching such that each edge is chosen at least once in this matching. Thus, one obtains a  $k$ -regular graph by replacing each edge by as many parallel edges, as its multiplicity in the  $b$ -matching. Berge [2] provided the following characterization of regularizable graphs. If  $G$  is connected and bipartite, then  $G$  is regularizable if and only if  $|N(U)| > |U|$  for each non-empty stable set  $U$  of  $G$ . This is a *strict* Hall condition for stable sets.

## 2 A characterization of equatable graphs

Which are the graphs  $G = (V, E)$  for which any initial assignment  $w : V \rightarrow \mathbb{N}_0$  is equatable? The following theorem provides the answer to that question.

**Theorem 1.** *Let  $G = (V, E)$  be a finite graph. The following statements are equivalent:*

1. *Every integer assignment  $w : V \rightarrow \mathbb{N}_0$  is equatable in  $G$ .*
2.  *$G$  is connected,  $|V|$  is odd and for all  $U \subseteq V$ , the graph  $G - U$  has less than  $|U|$  isolated vertices, where  $G - U$  is the vertex induced subgraph on  $V - U$ .*

Notice that condition 2) implies that  $G$  has at least 3 vertices. We will now provide the proof of this theorem. To do so, we rely on a well known result of Tutte that characterizes the existence of a perfect  $b$ -matching in a graph.

**Theorem 2 (Tutte [16]).** *Let  $G = (V, E)$  be a finite graph and let  $b : V \rightarrow \mathbb{N}_0$  be a weight function on the vertices of  $G$ . The following statements are equivalent.*

- a)  *$G$  has a perfect  $b$ -matching.*
- b) *For every (possibly empty) subset  $U$  of  $V$*

$$\sum_{x \in U} b(x) \geq \sum_{x \in I(U)} b(x) + S(G - U), \quad (2)$$

*where  $I(U)$  is the set of isolated vertices of  $G - U$  and  $S(G - U)$  is the number of connected components of  $G - U$  that are not isolated vertices whose total  $b$ -weight is odd.*

*Proof (Proof of Theorem 1).* Suppose that condition 2) holds. In order to show that any  $w \in \mathbb{N}_0^{|V|}$  is equatable, it is enough to show that each assignment  $w^{(v)}$  with

$$w^{(v)}(u) = \begin{cases} 1, & \text{if } u = v \\ 0, & \text{otherwise} \end{cases}$$

is equatable, since the corresponding steps decrease the weight of  $v$  relative to the other vertices by exactly one. We show this by establishing existence of a perfect  $b$ -matching with  $b(v) = 2 \cdot n$  and  $b(u) = 2 \cdot n + 1$  for each other vertex  $u \neq v$ .

Let  $U \subseteq V$ . We have to show (2). If  $U = \emptyset$ , then  $G - U = G$ . The total  $b$ -weight of  $G$  is even and  $G - \emptyset$  has only one component, since  $G$  is connected, thus  $S(G - U) = 0$ . Also,  $G - \emptyset$  does not have isolated vertices. This shows that the right-hand-side of (2) is 0.

If  $U \neq \emptyset$  then, by our assumption,  $|I(U)| \leq |U| - 1$ . We have

$$\sum_{x \in U} b(x) \geq (2n + 1)|U| - 1. \quad (3)$$

Indeed, there is equality in (3) only if  $v \in U$ .

On the other hand  $\sum_{x \in I(U)} b(x) \leq |I(U)|(2n + 1)$ . Therefore,

$$\sum_{x \in I(U)} b(x) \leq |I(U)|(2n + 1) \leq (|U| - 1)(2n + 1). \quad (4)$$

Finally, the term  $S(G - U)$  is at most the number of components of  $G$  that are not isolated vertices. Therefore,

$$S(G - U) \leq \frac{n - 1}{2}. \quad (5)$$

Inequality (2) is, therefore, satisfied because using (3), (4), and (5) inequality (2) reduces to

$$(2n + 1)|U| - 1 \geq (|U| - 1)(2n + 1) + \frac{n - 1}{2},$$

which clearly holds.

Suppose now that every  $w \in \mathbb{N}_0^{|V|}$  is equatable. Then clearly,  $G$  is connected. Also  $G$  has an odd number of vertices since the parity of the sum of weights is invariant under the edge-increment operation. In a graph with an even number of nodes, an equated assignment has even parity which shows that an odd initial assignment is not equatable.

Let  $U$  be any non-empty set of vertices of  $G$ . Assume to the contrary that  $G - U$  has  $k \geq |U|$  isolated vertices. Denote by  $I$  the set of isolated vertices in  $G - U$  and let  $v$  be a fixed vertex in  $U$ . Consider the weights assignment  $w : V \rightarrow \mathbb{N}_0$  such that  $w(v) = 1$  and for any other vertex  $v' \in V$  we have  $w(v') = 0$ . We reach a contradiction by showing that  $w$  is not equatable. To see

this observe that any step that increases by 1 the weight of a vertex in  $I$  must increase by 1 the weight of some vertex in  $U$ . It follows that at any moment the sum of the weights of the vertices in  $I$  is strictly smaller than the sum of the weights of the vertices in  $U$ . Because  $|I| \geq |U|$ , it is not possible to reach a situation where all vertices in  $I \cup U$  have the same weight.  $\square$

### 3 A polynomial-time algorithm to equate the weights

In this section, we deal with the computational problem of deciding whether an initial assignment  $w : V \rightarrow \mathbb{N}_0$  is equatable and, if so, how to compute a multiset of edges that leads to such equated weights. Let us recall the connection to the  $b$ -matching problem. If we know a number  $\beta \in \mathbb{N}$  such that all node-weights can be brought to  $\beta$  by increment-steps, then the multiset of edges leading to uniform weights  $\beta$  is a perfect  $b$ -matching with weights  $b(v) = \beta - w(v)$  for each  $v \in V$ . The primary question is then: Can  $\beta$  be efficiently computed? We will now give a positive answer to this question. The main result of this section is the following theorem. We first provide an upper bound on  $\beta$ .

**Theorem 3.** *Given a graph  $G = (V, E)$  and an integer weights assignment  $w : V \rightarrow \mathbb{N}_0$ , one can determine in strongly polynomial time whether  $w$  is positively equatable in  $G$ . Moreover, the smallest multiset of edges equating  $w$  can be determined efficiently.*

We again make use of Theorem 2. For some target value  $\beta \geq \max_{v \in V} w(v)$  let  $b_\beta : V \rightarrow \mathbb{Z}_{\geq 0}$  with  $b_\beta(v) := \beta - w(v)$ . By Theorem 2 there is a sequence of positive steps starting from node weights  $w$  and yielding uniform node weight  $\beta$  if and only if (2) holds for  $b = b_\beta$  and all subsets  $U$  of  $V$ .

**Lemma 1.** *If  $w$  is equatable, there is such a value  $\beta$  that is bounded from above by  $n \max_{v \in V} w(v)$ .*

*Proof.* For the trivial case where  $w$  is uniformly zero we can choose  $\beta = 0$ . Thus, in what follows we might assume that  $\max_{v \in V} w(v) \geq 1$ . Notice that with respect to (2) the only subsets  $U$  of  $V$  that might force  $\beta$  to be big are those with  $|U| > |I(U)|$  (otherwise, if  $|U| \leq |I(U)|$ , the left hand side of (2) as a function of  $\beta$  increases at most as fast as the right hand side does). For such subset  $U$ , however, and for  $\beta = n \max_{v \in V} w(v)$  we get

$$\begin{aligned} \sum_{x \in U} b_\beta(x) &\geq |U|(\beta - \max_{v \in V} w(v)) \\ &\geq |U|\beta - n \max_{v \in V} w(v) + n - |U| \\ &= (|U| - 1)\beta + n - |U| \geq \sum_{x \in I(U)} b_\beta(x) + S(G - U) \end{aligned}$$

which concludes the proof.  $\square$

*Proof (Proof of Theorem 3).* From now on we fix the parity of  $\beta$  (even or odd) such that  $S(G - U)$  no longer depends on the particular value of  $\beta$  (in our algorithm we deal with the two cases sequentially). In particular, for a fixed subset  $U$  of  $V$ , both the left hand side and the right hand side of (2) are linear functions of  $\beta$ . Therefore, for each  $U \subseteq V$ , one of three cases holds:

- (i) (2) is satisfied for all values of  $\beta$  or for no value of  $\beta$ ;
- (ii) there is a  $\beta_U \in \mathbb{Z}$  such that (2) is satisfied if and only if  $\beta \geq \beta_U$ ;
- (iii) there is a  $\beta_U \in \mathbb{Z}$  such that (2) is satisfied if and only if  $\beta \leq \beta_U$ .

This observation finally enables us to find the smallest feasible value of  $\beta$  (with fixed parity) by binary search in polynomial time: Let  $\alpha = \max_{v \in V} w(v)$  and  $\gamma = n \max_{v \in V} w(v)$ . Due to Lemma 1, we can restrict our search for a suitable value  $\beta$  to the interval  $[\alpha, \gamma]$ . For fixed  $\beta' \in [\alpha, \gamma]$ , we can test in polynomial time whether (2) is satisfied for all subsets  $U$  of  $V$  and obtain a violating subset  $U$  in the negative case. In fact such a violating set is found by the algorithm for the perfect  $b_{\beta'}$ -matching problem that also certifies the non-existence of a perfect  $b_{\beta'}$ -matching with a set  $U \subseteq V$  violating (2).

In the positive case, we can decrease the upper bound  $\gamma$  to  $\beta'$  and continue the search. In the negative case, we distinguish the three cases (i), (ii), and (iii) listed above w.r.t. the violating subset  $U$ . In case (i), there is no feasible  $\beta$  and we thus terminate the search. In case (ii) we obtain a new lower bound  $\beta_U > \beta'$  and thus continue the search after replacing  $\alpha$  with  $\beta_U$ . Finally, in case (iii) we obtain a new upper bound  $\beta_U < \beta'$  and thus continue the search after replacing  $\gamma$  with  $\beta_U$ .

Notice that the running time of the resulting binary search algorithm is only weakly polynomial. A strongly polynomial running time can be achieved by replacing binary search with *parametric search* [14].  $\square$

*Remark 1.* We sketch Megiddo's parametric search technique [14] in our setting. Suppose that we want to find the smallest even  $\beta$  such that there exists a perfect  $b_\beta$ -matching. Consider a fully combinatorial algorithm  $A$  for the non-parametric problem, that is, for the perfect  $b$ -matching problem. A fully combinatorial algorithm uses only additions, subtractions, and comparisons. In fact, such an algorithm exists for the perfect  $b$ -matching problem, if the parities of the  $b(v)$  are fixed, as it is for the case of all parametric  $b_\beta$ , if  $\beta$  is even, see [8, p. 186]. More precisely, the algorithm consists then of solving one general network flow problem and one perfect matching problem on graphs that are polynomial in the size of the graph  $G$  that is in our input, see, [8]. For both subproblems, there exist fully-combinatorial algorithms, for example the minimum mean-cycle algorithm [9] and Edmond's algorithm [5].

Algorithm  $A$  is now modified in order to solve the parametric problem. For this, the modified algorithm has to work with linear functions of the parameter  $\beta$  instead of just constant numbers. Notice that adding or subtracting two linear functions yields a linear function again. Comparing two linear functions, however, imposes a problem. Whenever algorithm  $A$  compares two numbers, the modified version first has to determine whether the desired value  $\beta_{OPT}$  is smaller or larger

than the unique point  $\beta^*$  at which the two linear functions cross (if at all). This can be decided by calling any algorithm  $B$  for the perfect  $b$ -matching problem as a subroutine for the fixed parameter value  $\beta^*$ . Depending on the outcome, the corresponding alternative of the comparison is chosen, for example in an if-conditional, and one continues to run algorithm  $A$  for the parametric problem. The number of calls of  $B$  is bounded by the number of comparisons performed by  $A$  which is strongly polynomial in the input size. In this way, finding the desired value  $\beta_{OPT}$  is reduced to a series of non-parametric  $b$ -matching problems.

## 4 Bipartite graphs.

Going back to the elementary puzzle presented at the beginning, observe that the corresponding graph  $G$  is bipartite where the two parts have the same cardinality. The initial weight  $w$  is not equatable, since the sum of the weights of the vertices in one part of the bi-partition is not equal to the sum of the weights.

Let  $G = (L, R, E)$  be a bipartite graph. An assignment of weights  $w$  to the vertices of  $G$  is called *balanced* if  $w(L) = w(R)$ , where for a subset  $U$  of vertices,  $w(U)$  is defined as  $\sum_{v \in U} w(v)$ .

We now characterize those bipartite graphs for which all balanced assignment  $w$  are equatable.

**Theorem 4.** *Let  $G = (L + R, E)$  be a bipartite graph. The following statements are equivalent:*

- i) *Every balanced assignment is positively equatable in  $G$ .*
- ii) *For any non empty set of vertices  $X$  that is properly contained either in  $L$  or in  $R$ ,  $|X| < |N(X)|$ .*

Here  $N(X)$  denotes the *neighborhood* of  $X$ , that is, the set of vertices in  $G$  that are neighbors of some vertex in  $X$ . Notice that condition ii) implies that  $|L| = |R|$  holds. Condition ii) is a “strict” version of the well known *Hall’s condition* for the existence of a perfect matching. A bipartite graph has a perfect matching if and only if for any non empty set of vertices  $X$  that is properly contained either in  $L$  or in  $R$ ,  $|X| \leq |N(X)|$ .

*Proof.* Suppose that every balanced assignment to the vertices of  $G$  is positively equatable. Assume to the contrary that there exists  $X \subset L$ ,  $0 < |X| < |L|$ , and  $|X| \geq |N(X)|$  (the symmetric case where  $X \subset R$  is similar). If  $N(X) = \emptyset$ , then the vertices in  $X$  are isolated and any balanced assignment of weights to the vertices in  $G$  where the vertices in  $X$  get weight 0 and some other vertex not in  $X$  gets a positive weight is not equatable.

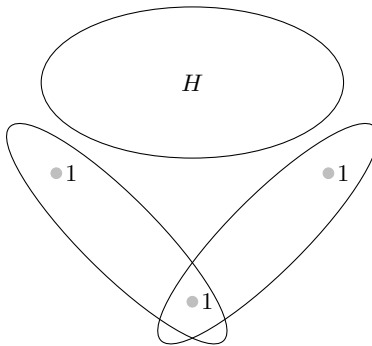
If  $N(X)$  is not empty, consider the following balanced assignment of weights to the vertices of  $G$ . Pick a vertex  $v \in L \setminus X$  and a vertex  $u \in N(X)$ . We define  $w(v) = w(u) = 1$  and for every other vertex  $z$  of  $G$  we define  $w(z) = 0$ . Clearly,  $w$  is balanced. However, the graph  $G$  together with the assignment of weights  $w$  is not equatable. This is because a positive step increases the total weight of the

vertices in  $N(X)$  by at least the same amount by which it increases the total weight of the vertices in  $X$ . If after a series of positive steps the weights of the vertices in  $G$  are the same, then in particular the total weight of the vertices in  $X$  is at least as large as the total weight of the vertices in  $N(X)$  (because  $|X| \geq |N(X)|$ ), but this is impossible because for the initial assignment of weight the total weight of the vertices in  $X$  is 0 while the total weight of the vertices in  $N(X)$  is 1.

Now, suppose that ii) holds. As in the proof of Theorem 1, it is enough to show that any assignment with  $w(u) = w(v) = 1$  for some  $u \in L$  and  $v \in R$  and  $w(x) = 0$  for any other vertex is equatable, since then, each balanced assignment can be equated. This however follows from the fact that  $G - \{u, v\}$  has a perfect matching, since it satisfies Hall's condition.  $\square$

#### 4.1 Hypergraphs

One can naturally generalize the equating problem to hypergraphs. In this setting, one is given a hypergraph  $H = (V, E)$  and an integer weights assignment  $w : V \rightarrow \mathbb{N}_0$ . A *positive step* on  $e \in E$  modifies  $w$  by increasing the weights of each  $u \in e$  by 1 unit. The rest of the definitions are generalized in the obvious way. Not surprisingly, deciding, whether one can equate the weights in a hypergraph is NP-complete. This follows by a reduction from *3-dimensional matching* [11].



**Fig. 2.** NP-completeness in the hypergraph setting.

Thus deciding whether a hypergraph has a perfect matching is an NP-complete problem. This can be trivially reduced to the equating problem by adding three new vertices and two new edges, each consisting of two of the three new vertices. The three vertices have weight 1 while all other vertices have weight 0. If these weights can be equated, then they all have weight 1 in an equated assignment. Thus, the weights can be equated if and only if the original hypergraph has a perfect matching. Consequently, the following theorem holds.



**Theorem 5.** *The decision problem of determining for any hypergraph  $H = (V, E)$  and any integer weights assignment  $w : V \rightarrow \mathbb{Z}$ , whether  $w$  is positively equatable in  $H$  is NP-complete.*

## References

1. Richard P Anstee. A polynomial algorithm for b-matchings: an alternative approach. *Information Processing Letters*, 24(3):153–157, 1987.
2. Claude Berge. Regularisable graphs i. *Discrete Mathematics*, 23(2):85–89, 1978.
3. W.H. Cunningham and A.B. Marsh. A primal algorithm for optimum matching. In *Polyhedral Combinatorics*, pages 50–72. Springer, 1978.
4. J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards*, 69:125–130, 1965.
5. J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.
6. Harold N Gabow. Data structures for weighted matching and nearest common ancestors with linking. In *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pages 434–443. Society for Industrial and Applied Mathematics, 1990.
7. Harold N Gabow and Robert E Tarjan. Faster scaling algorithms for general graph matching problems. *Journal of the ACM (JACM)*, 38(4):815–853, 1991.
8. A. M. H. Gerards. Matching. In M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, editors, *Network models*, volume 7 of *Handbooks in Operations Research and Management Science*, pages 135–224. North-Holland, Amsterdam, 1995.
9. Andrew V. Goldberg and Robert E. Tarjan. Finding minimum-cost circulations by canceling negative cycles. *J. ACM*, 36(4):873–886, October 1989.
10. John E Hopcroft and Richard M Karp. An  $n^2/2$  algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231, 1973.
11. R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, NY, 1972.
12. Alexander V Karzanov. On finding a maximum flow in a network with special structure and some applications. *Matematicheskie Voprosy Upravleniya Proizvodstvom*, 5:81–94, 1973.
13. Aleksander Madry. Navigating central path with electrical flows: From flows to matchings, and back. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 253–262. IEEE, 2013.
14. Nimrod Megiddo. Combinatorial optimization with rational objective functions. *Math. Oper. Res.*, 4(4):414–424, 1979.
15. Alexander Schrijver. *Combinatorial optimization. Polyhedra and efficiency (3 volumes)*. Algorithms and Combinatorics 24. Berlin: Springer., 2003.
16. WT Tutte. The factors of graphs. *Canad. J. Math*, 4(3):314–328, 1952.